ROS Jenkins CI/CD Pipeline Documentation

Pipeline Overview

This Jenkins pipeline automates the build, test, and deployment process for a ROS-based Autonomous Guided Vehicle (AGV) project.

Pipeline Stages

1. Cleanup Stage

- Purpose: Prepares a clean workspace for the build
- Actions:
  - Removes previous build artifacts
  - Ensures a pristine environment for each build

2. Checkout Stage

- Purpose: Retrieves the latest source code from GitHub
- Actions:
  - Clones the repository
  - Checks out the latest commit

3. Workspace Info Stage

- Purpose: Provides visibility into the current workspace
- Actions:
  - Lists workspace contents
  - Displays available disk space
  - Helps in debugging and understanding the build environment

4. Build Docker Image Stage

- Purpose: Creates a consistent build environment
- Details:
  - Uses `osrf/ros:noetic-desktop-full` as base image
  - Installs necessary ROS, Python, and system dependencies
  - Tagged as `ros-jenkins:91`
- Key Installed Packages:
  - ROS Noetic
  - Catkin tools
  - Python dependencies
  - Gazebo plugins
  - Development utilities

5. Build ROS Package Stage

- Purpose: Compile ROS packages

- Actions:
  - Initializes catkin workspace
  - Cleans previous builds
  - Builds `agv_sim` and `demo_pkg` packages
- Build Configuration:
  - Release mode
  - Verbose output
  - Summarized build results

6. Run Tests Stage

- Purpose: Execute package-level unit and integration tests
- Packages Tested:
  - `agv_sim`
  - `demo_pkg`
- Test Types:
  - ROS unit tests
  - Integration tests
- Reporting:
  - Generates XML test results
  - Captures test logs
  - Provides detailed test summary

7. Gazebo Simulation Deployment Stage

- Purpose: Run advanced simulation and performance testing
- Key Components:
  - Launches Gazebo simulation
  - Runs simulation tests
  - Captures rosbag data
  - Generates performance metrics
- Simulation Configuration:
  - Headless mode
  - Virtual framebuffer (Xvfb)
  - 60-second simulation duration
- Artifacts:
  - Simulation data bag
  - Performance report
  - Log summary

Environment Variables

- `DISPLAY`: Virtual display for headless simulation
- `ROS_MASTER_URI`: ROS master server configuration
- `ROS_HOSTNAME`: ROS network configuration
- `GAZEBO_RESOURCE_PATH`: Gazebo world resources

- `GAZEBO_MODEL_PATH`: Gazebo model resources

Best Practices Implemented

- Containerized build environment
- Comprehensive testing
- Detailed logging
- Performance tracking
- Artifact preservation

Troubleshooting

- Check Jenkins console output for detailed error messages
- Examine archived artifacts for simulation and test results
- Verify ROS and Gazebo configurations in source files

Future Improvements

- Add more comprehensive integration tests
- Implement performance benchmarking
- Enhance simulation scenario coverage